

(19)日本国特許庁 (J P)

(12) 公開特許公報 (A)

(11)特許出願公開番号

特開2000-163456

(P2000-163456A)

(43)公開日 平成12年6月16日(2000.6.16)

(51)Int.Cl. ⁷	識別記号	F I	テマコード [*] (参考)
G 0 6 F 17/50		G 0 6 F 15/60	6 6 4 P 2 G 0 3 2
G 0 1 R 31/28		11/22	3 1 0 A 5 B 0 4 6
G 0 6 F 11/22	3 1 0		3 4 0 A 5 B 0 4 8
	3 4 0	G 0 1 R 31/28	F 5 F 0 6 4
H 0 1 L 21/82		G 0 6 F 15/60	6 6 4 R 9 A 0 0 1

審査請求 未請求 請求項の数 3 O L (全 13 頁) 最終頁に続く

(21)出願番号 特願平10-334119

(22)出願日 平成10年11月25日(1998.11.25)

(71)出願人 000005108

株式会社日立製作所

東京都千代田区神田駿河台四丁目6番地

(72)発明者 中田 孝広

神奈川県秦野市堀山下1番地 株式会社日

立製作所汎用コンピュータ事業部内

(72)発明者 鈴木 薫

神奈川県秦野市堀山下1番地 株式会社日

立製作所汎用コンピュータ事業部内

(74)代理人 100080001

弁理士 筒井 大和

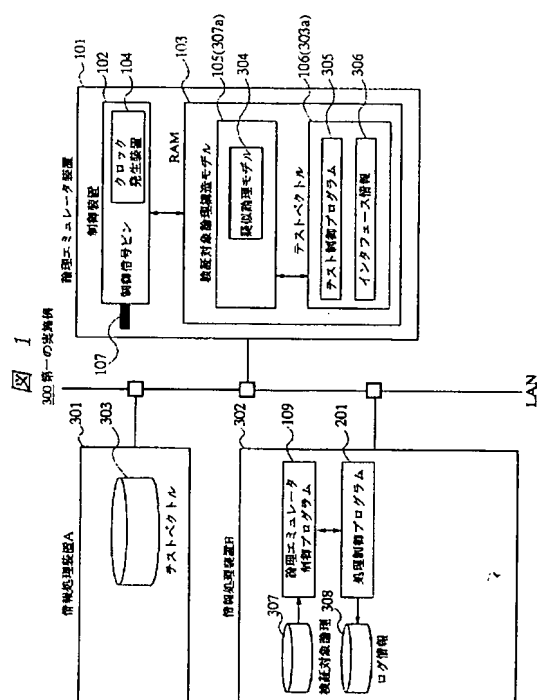
最終頁に続く

(54)【発明の名称】 論理検証方法

(57)【要約】

【課題】 操作性や観測性に優れた論理検証を、効率良く高速に行う。

【解決手段】 論理エミュレータ装置101に実装される、検証対象論理構造モデル105に当該検証対象論理構造モデル105のI/O環境等を提供する擬似論理モデル304を組み込み、テストベクトル303aには、当該テストベクトル303aの検証対象論理構造モデル105への入力による論理検証動作を制御するテスト制御プログラム305およびインタフェース情報306を組み込み、外部の汎用言語で記述されたGUIインタフェースを持つ処理制御プログラム201から、インタフェース情報306を介して、擬似論理モデル304およびテスト制御プログラム305との間でテスト情報の授受を行うことで、GUIインタフェースにて論理検証の操作性と観測性を向上させ、高速な論理エミュレータ装置101による効率の良い大規模論理装置の一貫した論理検証を達成する。



【特許請求の範囲】

【請求項1】 検証対象の目的論理装置の論理構造モデルが実装された論理エミュレータまたは前記目的論理装置の実チップが実装された専用試験装置における論理検証に用いられるテストベクトルの中に、前記論理エミュレータまたは前記専用試験装置の外部に設けられた処理制御プログラムとの間で情報の授受を行うことで、前記テストベクトルの入力による前記論理検証を前記処理制御プログラムから制御可能にするテスト制御プログラムを実装することを特徴とする論理検証方法。

【請求項2】 請求項1記載の論理検証方法において、前記論理エミュレータでは、前記目的論理装置と等価な論理機能を実現するための第1の論理構造モデルと、前記第1の論理構造モデルの動作環境を提供する論理機能の少なくとも一部を実現するための第2の論理構造モデルとを実装し、前記目的論理装置の実行環境を実現するための外部ハードウェアとの接続を必要とすることなく、前記目的論理装置の論理検証を行うことを特徴とする論理検証方法。

【請求項3】 請求項1または2記載の論理検証方法において、前記処理制御プログラムは、前記テスト制御プログラムとの間における情報の授受を行う機能、前記論理エミュレータを制御する論理エミュレータ制御プログラムとの間における情報の授受を行う機能、前記目的論理装置の論理検証をソフトウェアにて行う論理シミュレータとの間における情報の授受を行う機能、前記専用試験装置との間における情報の授受を行う機能、前記テスト制御プログラム、前記論理エミュレータ制御プログラム、前記論理シミュレータ、前記専用試験装置の少なくとも一つとの間で授受される前記情報の可視化表示やユーザ入力の受け付けを行う汎用グラフィカル・ユーザ・インタフェース、のうちの少なくとも一つを備えたことを特徴とする論理検証方法。

【発明の詳細な説明】

【0001】

【発明の属する技術分野】本発明は、論理検証技術に関し、特に、論理シミュレータや論理エミュレータを用いた論理回路装置の機能検証技術等に適用して有効な技術に関する。

【0002】

【従来の技術】従来の論理エミュレーション装置（論理エミュレータ）を用いた論理エミュレーションは、論理検証対象のシステム装置や論理チップの一部分である論理モデルを動作記述言語で動作を記述し、設計された論理構造モデルを論理コンパイルして論理エミュレーション装置内のRAMに実装し、論理モデル動作をエミュレ

ーションするインサーキット方式で論理検証を行っていた。また、インサーキットではない装置や論理チップは、インサーキット論理とのインタフェースを確保するために論理エミュレーション装置の外部接続ピンに互いの外部信号線を接続し、信号値レベルで動作の同期をとっている。また、論理動作の観測やテストベクトルを個別に指定する場合には、論理エミュレーション装置内の観測可能な内部信号や外部ピンに対して論理値を与えたり、あらかじめ指定された信号線に対して信号線単位での観測を行い、論理動作の確認を行っていた。

【0003】このような論理エミュレーション技術に関しては、特開平02-245831号公報に示されたような方法等が知られている。すなわち、検証対象の論理機能を外部からプログラム可能なゲートアレイ上にマッピングすることにより、目的の論理機能をハードウェア的に実現して、高速な実行および論理検証を可能にしようとするものである。

【0004】

【発明が解決しようとする課題】従来のインサーキット方式は、近年の論理規模の増大や論理の複雑さに対して部分論理での論理エミュレーションのためシステムテストのような論理品質を確保するには充分とは言えず、論理チップ全体での論理エミュレーションによるシステム論理検証が必要になってきている。論理チップ全体での論理検証では、論理規模の増大や論理の複雑化も進み、従来の信号線レベルでの論理値を与えたり、信号線の値を観測したりする論理エミュレーションでは信号線全てに論理値を設定しなければならないため、効率の良い論理検証を行うことが難しい。観測性や操作性を向上させるには、マンマシンインタフェースを充実させ、テスト容易性を向上させる必要がある。

【0005】また、論理チップ全体での論理シミュレーションは、論理シミュレータではソフトウェアで論理モデルを作成するため、ハードウェア動作に比べてソフトウェア動作のため論理検証に莫大な時間を要し、製品を短期開発し、早期出荷しなければならない今日では適用が難しく、解決しなければならない課題の一つである。

【0006】また、検証対象の論理が周辺装置などの論理の場合には、論理単体での論理検証動作を確認出来ない場合もあり、検証レベルの異なるアーキテクチャ論理シミュレータ等を併用して検証対象論理の論理検証を行ったり、専用の論理シミュレータを用いたりして論理検証を行うことが必要になる。したがって、効率の良い論理シミュレータを選択し、システムテストレベルで併用出来る論理エミュレーション方式の構築が重要な課題である。

【0007】また、論理エミュレーションが終了した段階で実チップが製造され、チップ単体または装置全体での論理品質検査を実施する時、それまでの論理検証環境を継続して使用できれば論理不良が発見された場合に論

理シミュレーションや論理エミュレーションへのフィードバックが容易であり、検証環境の構築ということを考えれば一貫した論理検証環境の構築が必要である。

【0008】本発明の目的は、論理チップ全体での論理エミュレーションによるシステム論理検証を短時間に効率よく行うことが可能な論理検証技術を提供することにある。

【0009】本発明の他の目的は、マンマシンインタフェースを充実させ、論理エミュレーションにおける内部状態の観測性や操作性、さらにはテスト容易性を向上させることが可能な論理検証技術を提供することにある。

【0010】本発明の他の目的は、論理モデルから実チップに至るまでの一貫した論理検証を実現することが可能な論理検証技術を提供することにある。

【0011】本発明の他の目的は、論理検証の環境構築に要する工数や期間を短縮して、論理検証工程におけるコスト削減を実現することが可能な論理検証技術を提供することにある。

【0012】

【課題を解決するための手段】本発明は、目的論理装置の論理検証を目的として、論理エミュレータ上に目的論理装置の論理構造モデルと、論理構造モデルと共に動作することで当該論理構造モデルの実行環境を提供する疑似論理モデルを実装し、論理エミュレータ上で実行されるテストベクトルには、外部の処理制御プログラムとの間で情報の授受を行うことで論理検証動作を制御するテスト制御プログラムを実装するものである。

【0013】疑似論理モデルとプログラムと論理エミュレータを制御する処理制御プログラムは、たとえばネットワークを介して論理エミュレータに接続された情報処理装置上に実装され、論理エミュレータと当該処理制御プログラムとの間で情報通信を行う手段と、論理エミュレータを非同期に動作させる手段と、各種情報の可視化表示や情報の入力環境等を提供するグラフィカルユーザインタフェースと、論理エミュレータ以外の論理シミュレータ等の検証手法の異なる論理検証プログラムを当該処理制御プログラムに接続する接続プラグインタフェースと、論理エミュレータ上で動作するテスト制御プログラムやテストベクトルとのインタフェースとを有しており、目的論理装置の論理検証の操作性と観測性を向上させ効率よく高速に論理検証を可能とする。

【0014】また、目的論理装置の実チップが実装される専用試験装置において、実チップに入力されるテストベクトル内に、外部の処理制御プログラムとの間で情報の授受を行うことで論理検証動作を制御するテスト制御プログラムを実装するものである。この場合、処理制御プログラムは、上述の構成の他に、任意の接続インタフェースを介して専用試験装置が接続される情報処理装置に実装され、前記接続インタフェースを制御するデバイスドライバとの情報の授受を行うドライバ制御プログラ

ムを有し、実チップの論理検証における操作性や観測性を向上させ効率良く高速に論理検証を行う。また、テストベクトルや、テスト制御プログラム、処理制御プログラムは、論理エミュレータの場合と共通のものをを用いることができる。

【0015】また、テストベクトル内のテスト制御プログラムと疑似論理モデルとのインタフェースを持つ処理制御プログラムにおいて、論理エミュレータに実装した目的論理装置の論理モデルの内部信号値の観測や制御を行い、処理状態を1/0リクエストとして制御し、テスト制御プログラムと1/0リクエストを処理制御プログラムが送受信することで、論理エミュレータを制御しながら目的論理装置の論理検証を行うことができる。

【0016】

【発明の実施の形態】以下、本発明の実施の形態を図面を参照しながら詳細に説明する。

【0017】図1は本発明の第一の実施の形態である論理検証方法が実施される情報処理システムの構成の一例を示す概念図であり、図2は、本発明の論理検証方法を実現するためのソフトウェアの構成の一例を例示した概念図である。また、図8は、本実施の形態の論理検証方法の参考技術の論理エミュレーションシステムの構成の一例を示す概念図である。

【0018】先ず、図8を用いて本実施の形態の説明の前に、本実施の形態の参考技術の論理エミュレーションシステムの概要を説明する。

【0019】図8の論理エミュレーションシステム100は、LAN接続された論理エミュレータ装置101と、同じくLAN接続された情報処理装置108上で動作する論理エミュレータ制御プログラム109で構成される。論理エミュレータ装置101は、制御装置102と読み書き可能なRAM103で構成され、制御装置102には論理エミュレータ装置101に供給される動作クロックを発生させるクロック発生装置104と、論理エミュレーション制御用の制御信号ピン107などで構成される。なお、本発明に関する構成要素のみにについて特化して説明を行う。

【0020】制御信号ピン107は、エミュレーション状態を論理エミュレータ制御プログラム109に報告したり、動作指示を受け取るための外部ピンである。RAM103は、検証対象論理構造モデル格納エリア105に後述の検証対象論理構造モデル114aを実装したり、テストベクトル格納エリア106にテストベクトルを実装するために用いる。論理エミュレータ制御プログラム109は、論理エミュレータ装置101および論理エミュレータ制御プログラム109を制御するための制御部110と、RAMアクセス部112と、動作記述言語で記述された検証対象論理114を論理コンパイルする論理コンパイラ111と、論理エミュレーションシステム全体をグラフィカルに表示、制御するグラフィカル

ユーザインタフェース（GUI）制御部 113 で構成される。論理コンパイラ 111 は、検証対象論理 114 を入力とし、論理コンパイルして検証対象論理構造モデル 114a として論理エミュレータ装置 101 内の RAM 103 の検証対象論理構造モデル格納エリア 105 に実装する。

【0021】次に論理エミュレーションを実際に行う手順について説明する。前述した検証対象論理 114 を論理コンパイルして検証対象論理構造モデル 114a として RAM 103 に実装した後、テストベクトルを RAM 103 のアクセス部 112 によりテストベクトル格納エリア 106 に実装し、制御部 110 によって制御信号ピン 107 に対して動作条件情報を送り、クロック発生装置 104 によって実際にクロックが供給され、論理エミュレータ装置 101 を動作させ論理エミュレーションを行う。次に一定サイクル論理エミュレーションを行った後、制御装置 102 は、制御信号ピン 107 にエミュレーション動作状態情報をセットし、この情報を制御部 110 が監視し、その状態情報に合わせて引き続きクロックを供給したり、中断して RAM 103 の状態を観測したりする。これらの処理を繰り返し行うことで論理検証を行っている。

【0022】次に図 2 に本発明の各実施の形態の論理検証方法を実現するための処理制御プログラムの一例を示す。処理制御プログラム 201 は、ネットワーク上に起動された論理エミュレータ制御プログラム 109 との間でプロセス間通信を行うプロセス間通信制御部 203 と、論理エミュレータ制御部 204 と、テストベクトル制御部 205 と、論理エミュレータ装置 101 との間で送受信されるデータの変換処理を行うデータ変換部 206 と、論理エミュレータ装置 101 以外の検証装置および検証システムを処理制御プログラム 201 に接続するための接続プラグインタフェース 207 と、本発明の論理検証方式を総合的にコントロールするグラフィカルユーザインタフェース（GUI）制御部 202、で構成される。なお、詳細な各処理部の説明は実施の形態の説明で行う。また、図 2 には、簡単のため、プロセス間通信制御部 203～接続プラグインタフェース 207 の構成要素をすべて含む構成が例示されているが、これらのうちの必要な構成要素のみを含む構成も本発明に含まれる。

【0023】次に図 1 を用いて本発明の第一の実施の形態である論理検証方法が実施される論理エミュレーションシステムを説明する。図 1 に示した第一の実施の形態の論理エミュレーションシステム 300 において、検証対象論理を機能レベルアーキテクチャを実現した論理として論理コンパイルして論理エミュレータ装置 101 の RAM 103 にセットし、論理検証を行う手法を説明する。

【0024】なお、本実施の形態の論理エミュレーション

ンシステム 300 では、論理エミュレータ装置 101 は、インサート方式の論理検証において、後述の擬似論理モデル 304 を論理エミュレータ装置 101 内に実装することにより、検証対象の論理構造モデルのみを実装した状態でのベクトル検証を可能にするものであり、検証対象の論理モデル（と等化な動作を行う実チップ）の実際の動作環境を提供するために周辺のハードウェア回路を論理エミュレータ装置 101 に接続する必要はない。

【0025】たとえば検証対象がマイクロプロセッサである場合、その全体の論理機能の検証には、実際のシステムの構築に用いられる周辺回路との入出力環境を実現する必要があるため、図 8 に例示されたような参考技術の論理エミュレーションシステム 100 では、あらかじめ、周辺回路のハードウェア環境を準備しておき、マイクロプロセッサの動作をエミュレートする論理エミュレータ装置 101 に対して配線接続する必要がある。これに対して、本実施の形態の論理エミュレーションシステム 300 の場合には、後述の擬似論理モデル 304 が、目的の検証対象論理構造モデル 307a の動作に必要な I/O インタフェースを実現するので、実際の周辺回路等への接続（インサート接続）は不要であり、論理エミュレータ装置 101 の単体で検証動作が可能である。

【0026】本実施の形態では、テストベクトル 303 は、LAN に接続された情報処理装置 301（A）に格納されている。また、LAN に接続された情報処理装置 302（B）には、論理エミュレータ制御プログラム 109 および処理制御プログラム 201 が実装されて実行される。情報処理装置 301（A）や情報処理装置 302（B）は、たとえば UNIX 等の OS で動作するワークステーションやパーソナルコンピュータ等で構成される。

【0027】先ず、前準備として検証対象論理 307 とテストベクトル 303 を準備する。前記の検証対象論理 307 には、処理制御プログラム 201 とのインタフェース機能を有する擬似論理モデル 304 を組み込み、論理コンパイラ 111 で論理コンパイルして検証対象論理構造モデル 307a としておく。擬似論理モデル 304 の機能としては、論理動作中の内部信号線の論理値を観測したり監視したりする機能と、検証対象論理 307

（検証対象論理構造モデル 307a）の動作をコントロールするシステムコントロール機能と処理制御プログラム 201 とのインタフェース機能を有する。また、テストベクトル 303 には、処理制御プログラム 201 とのインタフェース機能を持つテスト制御プログラム 305 と、処理制御プログラム 201 とのインタフェースに使用するインタフェース情報 306 を組み込んでおく。

【0028】次に、論理エミュレータ制御プログラム 109 を起動し、準備した検証対象論理 307 のコンパイ

ル結果を、RAMアクセス部112を用いて、検証対象論理構造モデル307aとしてRAM103内の検証対象論理構造モデル格納エリア105にセットする。

【0029】次に処理制御プログラム201を起動し、準備しておいたテストベクトル303をデータ変換部206でRAM103にセット可能な形式のテストベクトル303aに変換し、論理エミュレータ制御部204内のRAMアクセス部210経由でRAM103内のテストベクトル格納エリア106にセットする。

【0030】図8に示した参考技術の論理エミュレーションシステム100と異なる点は、擬似論理モデル304とテストベクトル303a内のテスト制御プログラム305およびインタフェース情報306、および論理エミュレータ制御プログラム109と独立な処理制御プログラム201が設けられていることである。

【0031】また、本実施の形態の論理エミュレータ制御プログラム109は、Tcl/Tkインタフェースを有するため、処理制御プログラム201をTcl/Tkで作成することにより、参考技術である図8の論理エミュレータ制御プログラム109の機能をそのまま使用できると共に論理エミュレータ制御プログラム109の拡張機能の一貫として容易に接続可能である。つまり、処理制御プログラム201は、論理エミュレータ装置101を外部から操作可能にする拡張プログラムと、

【0032】なお、Tcl/Tkとは、カリフォルニア大学バークレー校で開発されたグラフィカルインタフェース（GUI）を有する汎用スクリプト言語であり、オープンシェルスクリプトとして多くの大学や研究所の他、大小様々な企業で利用されている。

【0033】また、本発明で採用したTcl/Tkインタフェースは、本実施例の論理エミュレータ装置がTcl/Tkインタフェースを有していたためであり、本発明を実施する際、マンマシンインタフェースを提供するGUI機能はXウィンドウシステムやMotif等で実現することが可能である。更に言い換えれば、GUIインタフェースを持ったシステムであれば、その機能やシステムを限定するものではない。

【0034】次に実際に論理エミュレーションを行う手順に沿って説明する。テストベクトル303aがセットされた後、設定情報制御部215により詳細な論理エミュレーション動作の設定やテスト制御プログラム305への設定、インタフェース情報306の詳細設定、動作周波数など論理エミュレーション動作に必要な各種設定を行う。これらの設定は、GUI制御部113やGUI制御部202によって制御されている。各種設定が完了した後、論理エミュレータ制御部204内の動作制御部209が擬似論理モデル304に対してリセット要求を出し、検証対象論理構造モデル307aの状態を動作可能な状態にする。この後、制御信号ピン107に対して動作クロック数を与え、制御装置102はクロック発生

装置104により、指定されたクロック数分だけクロックが論理エミュレータ装置101に供給され実際に動作する。指定された一定クロック数分だけ動作しているとき、テスト制御プログラム305はインタフェース情報306に対して、内部動作状態やテスト動作状態などをセットする。また、指定されたクロック数に到達することなく、エミュレーションを中断したり、処理制御プログラム201に対してメッセージ等の出力要求がテスト制御プログラム305で発生した場合には、その状態情報をインタフェース情報306にセットする。

【0035】擬似論理モデル304は、インタフェース情報306を監視しているため、これらの要求を検出すると、制御装置102の制御信号ピン107に対して、トリガーイベントを発生させ制御装置102に中断報告を行う。このように擬似論理モデル304は、エミュレーション動作を中断させる機能を有し、処理制御プログラム201と直接対話型で情報のやりとりが可能である。つまり、論理内部の内部信号の値を監視したりすることで、エミュレーションを中断し、その情報を処理制御プログラム201に伝達可能である。

【0036】次に、処理制御プログラム201のトリガーイベント制御部208が、制御信号ピン107の監視をしているため、トリガーイベントを検出すると中断情報の格納されているインタフェース情報306をRAMアクセス部210経由で取得し、取得した情報は、表示するためにデータ変換部206で変換し、表示部212に引き渡し表示を行う。中断情報に対して、テストベクトル制御部205で取得情報の解析を行い、テスト制御プログラム305が入力またはコマンド等を要求している場合には、入力制御部213により、オペレーターのキー入力やGUIウインドウ操作を受け、入力された情報をインタフェース情報306にセットする。セットした後、クロックを供給すれば論理エミュレーションは継続される。

【0037】なお、これらのエミュレーション結果や動作中の状態を保存する場合には、ログデータ採取制御部216により、表示された情報や入力された情報、およびアクセス可能なRAM情報などを取得し、ログ情報308として保存可能である。また、処理制御部211とウインドウ制御部214は、一貫して処理制御プログラム201のコントロールを行い、各種操作をサポートする。

【0038】以上のようにGUI制御部202の各種機能とテストベクトル制御部205を用いてRAM103内のテスト制御プログラム305と対話的に論理エミュレーションを行うことで論理エミュレータ装置101内の専用の擬似論理モデル304との対話手段を確保することが可能となり、論理エミュレーション動作において、インタフェース情報306のみの操作で論理エミュレーションの制御が可能であり、操作性や観測性を向上

させ、効率の良い論理検証が可能である。また、これらの手法は、論理エミュレータ制御プログラム 109 を間接的に制御しているため、図 8 に例示された参考技術を包含し、その上で外部から論理エミュレータ装置 101 を制御可能である。更に、テスト制御プログラム 305 で論理エミュレーション動作をコントロール可能であるため、論理エミュレーションを中断させることを最小限にし、論理エミュレータ装置 101 の高速性を最大限に利用できるため、高速な論理エミュレーションが可能である。

【0039】次に第一の実施の形態の論理エミュレーションにおける具体的な処理フローの一例を図 3 に示す。図 3 では、論理エミュレーション動作途中でのテスト制御プログラム 305 がメッセージ出力要求とコマンド入力要求を発生したときの本実施の形態の構成部分の作用についてのみの処理フローを説明する。まず、図 3 での処理ステージは、オペレータ 401、処理制御プログラム 201、擬似論理モデル 304、テスト制御プログラム 305 のステージに区分し、これら各ステージの連携動作を処理フローとして示す。

【0040】前述の検証対象論理 307 とテストベクトル 303 が、それぞれ検証対象論理構造モデル 307 a およびテストベクトル 303 a として RAM 103 にセットされ、クロックを供給されればエミュレーションが開始可能な状態である場合において、オペレータは動作開始指示を行う（ステップ 402）。このステップ 402 の動作開始指示には、動作クロック数（動作サイクル数）が指示されている。次に、処理制御プログラム 201 はエミュレーション開始を実際に行い（ステップ 403）、論理エミュレータ装置 101 は論理エミュレーションを開始する。論理エミュレーションが開始されると擬似論理モデル 304 とテスト制御プログラム 305 が動作を開始する（ステップ 404）。この時、擬似論理モデル 304 はインタフェース情報 306 の監視を常に行っている（ステップ 405）。そして、テスト制御プログラム 305 が動作中にメッセージをオペレータに報告するため、メッセージ出力要求が発生し、更にそのメッセージに対する応答コマンドの入力要求が発生した場合（ステップ 406）、テスト制御プログラム 305 はインタフェース情報 306 に出力メッセージと擬似論理モデル 304 が検出可能なイベント情報をセットする（ステップ 407）。次に擬似論理モデル 304 はインタフェース情報 306 の監視によりイベントの検出を行い（ステップ 408）、制御信号ピン 107 に対してトリガーイベントを発生させる（ステップ 409）。トリガーイベントが発生すると制御装置 102 はエミュレーションを中断し（ステップ 410）、処理制御プログラム 201 は中断状態からトリガーイベントによる中断であることを認識し（ステップ 411）、インタフェース情報 306 を RAM 読み出しにより取得する（ステップ

412）。取得した情報はデータ変換部 206 を通してデータ変換し（ステップ 413）、テスト制御プログラム 305 からの出力メッセージとして表示部 212 のウィンドウに表示する（ステップ 414）。通常は、メッセージの出力のみであれば、表示後エミュレーションを再開するが、この実施の形態の場合には入力コマンド要求も同時に発生しているため、コマンドの入力要求をオペレータ 401 に対して行い（ステップ 415）、処理制御プログラム 201 は入力待ち状態になる（ステップ 417）。オペレータ 401 は、キー入力を行い（ステップ 416）、テスト制御プログラム 305 に対するコマンドを入力する。次にキー入力されたコマンドを取得し（ステップ 418）、データ変換部 206 でデータ変換を行い（ステップ 413）、インタフェース情報 306 にセットすべく、RAM 書き込みを行う（ステップ 419）。次に、エミュレーション動作を再開し（ステップ 420）、擬似論理モデル 304 とテスト制御プログラム 305 は動作を再開する（ステップ 421）。再開後は、インタフェース情報 306 から入力コマンドを取得し（ステップ 422）、そのコマンドに合わせた動作を行う（ステップ 423）。

【0041】このように、オペレータ 401 とテスト制御プログラム 305 の間でインタフェース情報を相互に転送し、対話的に論理エミュレーションを行うことで、オペレータは外部から論理エミュレータ装置 101 を制御し、テスト制御プログラム 305 は内部から論理エミュレータ装置 101 を制御することで効率の良い論理検証が可能である。なお、テスト制御プログラム 305 と同様にインタフェース情報 306 を監視する擬似論理モデル 304 も論理エミュレータ装置 101 を内部から制御可能であり、特に論理モデルであるため、内部信号などの観測性に優れ、擬似論理モデル 304 はハードウェア的に、テスト制御プログラム 305 はソフトウェア的に、論理エミュレータ装置 101 を制御可能であり、効率の良い論理エミュレーションが可能である。

【0042】すなわち、本実施の形態によれば、テストベクトル格納エリア 106 のテストベクトル 303 a 内に設けられたテスト制御プログラム 305 や、擬似論理モデル 304 によりテストベクトル 303 a とのインタフェースを確保し、テストベクトル 303 a からの I/O リクエストを処理制御プログラム 201 が送受信し、Tcl/Tk 等の汎用の GUI 機能を用いて操作性や観測性を向上させることにより、論理エミュレーションでのシステム論理検証を効率良く高速に行うことが可能である。

【0043】次に本発明の第二の実施の形態である論理検証方法が実施される論理エミュレーションシステム 500 を図 4 を参照して説明する。図 4 の論理エミュレーションシステム 500 では論理エミュレータ装置 101 の代わりに専用情報処理装置などを接続し、検証論理チ

ップ508の論理検証を行う手法を説明する。構成としては、LAN接続された情報処理装置301(A)は第一の実施の形態と同じでテストベクトル303も同じものである。また、情報処理装置501(C)は、この第二の実施の形態では一例として、たとえば汎用のパーソナルコンピュータ用OSであるWindows系OSで動作するパーソナルコンピュータとして説明する。情報処理装置501(C)は、Tcl/Tkインタプリタ503と、処理制御プログラム201と、ドライバ制御プログラム601と、デバイスドライバ505で構成される。

【0044】そして、情報処理装置CにパラレルI/Oインタフェース506で接続される専用情報処理装置502は、制御装置507と、検証論理チップ508と、RAM509で構成される。ここでの検証論理チップ508は、第一の実施の形態での検証対象論理構造モデル307a(検証対象論理307)を実際のチップとして製造したものである。そして、RAM509のテストベクトル格納エリア106には、第一の実施の形態で使用したテストベクトル303a(テストベクトル303)をそのままセットする。

【0045】デバイスドライバ505は、Windows系OSのデバイスドライバでパラレルI/Oインタフェース506を制御可能なデバイスドライバである。そして、このデバイスドライバ505を制御するためのドライバ制御プログラム601は、処理制御プログラム201の接続プラグインタフェース207とのインタフェースを持つことにより、処理制御プログラム201とデバイスドライバ505とを接続し、処理制御プログラム201がシステム全体を制御可能とする。そして、デバイスドライバ経由で制御装置507を制御することで専用情報処理装置502を制御する。

【0046】また、Tcl/Tkインタプリタ503は、Windows系OS版のTcl/Tkのインタプリタであり、処理制御プログラム201は、第一の実施の形態のUNIX系OS版のものがそのままWindows系OS版でも動作可能である。

【0047】したがって、この第二の実施の形態の検証目的は、実機に搭載する論理チップが製造された時点で、論理エミュレーションで用いたテストベクトル303と処理制御プログラム201を用いて、論理チップの論理品質を検証することである。つまり、実機システム検証の前に論理チップ単体の論理品質検証を行うことである。

【0048】次に、ドライバ制御プログラム601の概要を図5に示す。ドライバ制御プログラム601は、制御部602と、処理制御プログラムインタフェース部603と、デバイスドライバ制御部604と、ログデータ採取制御部605で構成される。制御部602は、ドライバ制御プログラム601の全体の制御を行い、処理制

御プログラムインタフェース部603は、処理制御プログラム201からのRAMアクセスや専用情報処理装置502の制御指示を制御する。デバイスドライバ制御部604は、デバイスドライバ505とのインタフェースを持ち、デバイスドライバ経由でパラレルI/Oインタフェース506を制御する。ログデータ採取制御部605は、ドライバ制御プログラム601で取得可能なログデータを採取し、ログ情報504として出力あるいは保存する機能を持つ。

【0049】次に第二の実施の形態の具体的な処理フローを図6に示し説明する。図6では、専用情報処理装置502が動作途中でのテスト制御プログラム305がメッセージ出力要求とコマンド入力要求が発生したときの本発明の部分に着目して処理フローを説明する。先ず、図6での処理ステージは、オペレータ701、処理制御プログラム201、ドライバ制御プログラム601、テスト制御プログラム305のステージに区分されており、全体処理フローを表している。

【0050】前述の検証論理チップ508が専用情報処理装置502にセットされ、テストベクトル303aがRAM509のテストベクトル格納エリア106にセットされ、動作開始指示待ち状態である場合において、オペレータ701は検証開始指示を行う(ステップ702)。次に処理制御プログラム201は、ドライバ制御プログラム601に対して動作開始指示を行い(ステップ703)、ドライバ制御プログラム601は、専用情報処理装置502を作動させる(ステップ704)。装置が作動するとテスト制御プログラム305は動作を開始する(ステップ705)。また、処理制御プログラム201は、ステップ703を行った後、インタフェース情報の監視を開始する(ステップ706)。インタフェース情報306の監視では、RAM509のポーリングアクセスのため、ドライバ制御プログラム601は、常にRAMアクセスを行い、RAM読み出しを行っている(ステップ707)。これらの状態であるとき、テスト制御プログラム305は、メッセージの出力要求とコマンド入力要求が発生すると(ステップ708)、メッセージとイベントをインタフェース情報にセットする(ステップ709)。次にインタフェース情報306の監視を行っている処理制御プログラム201が、このイベントを要求として検出し(ステップ710)、インタフェース情報306を取得する(ステップ711)。取得したデータはデータ変換され(ステップ712)、メッセージとしてウインドウに表示される(ステップ713)。表示後は、オペレータ701がコマンド入力要求(ステップ714)に対してキー入力やウインドウ操作(ステップ715)を行う。

【0051】次に入力待ち状態の処理制御プログラム201は(ステップ716)、入力されたキー入力を取得し(ステップ717)、データ変換を行い(ステップ7

12)、インタフェース情報306としてRAM509に書き込むことにより(ステップ718)、テスト制御プログラム305に転送する(ステップ719)。転送後は処理制御プログラム201は再度、インタフェース情報306の監視状態に入る(ステップ706)。テスト制御プログラム305は、前記ステップ708の要求を出した後も動作中であり(ステップ720)、転送されたインタフェース情報306から入力イベントを検出すると(ステップ721)、インタフェース情報306から入力コマンドを取得し(ステップ722)、取得したコマンドに対する動作を行う(ステップ723)。このような一連の動作を行い、対話的に検証論理チップ508の論理検証を行う。

【0052】また、第一の実施の形態と共通したテストベクトル303と処理制御プログラム201をそのまま使用可能であり、検証対象論理構造モデル105を用いた論理エミュレーションと、検証論理チップ508等の実チップの単体検証を同じ環境で行うことが可能であり、論理モデルから実チップに至るまでの一貫した論理検証が可能となる。

【0053】すなわち、論理エミュレータ装置101を用いた論理エミュレーションによる論理検証工程と、検証論理チップ508等の実チップによる論理検証工程に共通のソフトウェアやテストベクトルを使用できることで、各工程別にソフトウェアやテストベクトルを用意する等の重複した労力を軽減でき、効率の良い論理検証が可能となる。

【0054】次に本発明の第三の実施の形態である論理検証方法が実施される論理エミュレーションシステム800を図7に示す。図7の論理エミュレーションシステム800では図1の第一の実施の形態に論理シミュレータを加え、論理検証手法と論理検証レベルの異なる論理エミュレータと論理シミュレータが混在する論理検証方式の場合が例示されている。

【0055】この第三の実施の形態としては、具体的には、たとえば論理シミュレータ802でマイクロプロセッサ等の論理動作を行わせ、論理エミュレータ装置101では、このマイクロプロセッサを含むシステムで使用されるI/O制御デバイス等の論理を実装することで、双方の実チップを使用したシステムを組み上げる前に、両者の連携した動作における双方の論理検証を行う場合が考えられる。

【0056】この第三の実施の形態での論理シミュレータ802は機能レベル命令シミュレータであり、論理シミュレータ802をマイクロ命令動作として説明する。情報処理装置801(D)の上で、論理エミュレータ制御プログラム109と、処理制御プログラム201と、論理シミュレータ802を起動する。この3つのプログラムの間では、処理制御プログラム201の接続プラグインタフェース207によって論理シミュレータ802

が接続され、プロセス間通信制御部203によってプロセス間通信を行う。よってシステム全体を統括し、コントロールしているのは処理制御プログラム201である。

【0057】論理検証の手順としては、第一の実施の形態の要領で論理エミュレータ装置101を作動させ動作可能状態とする。次に論理シミュレータ802を処理制御プログラム201が起動し、論理シミュレータ802に与えるテストベクトル(テスト命令列)806をRAMモデル805にセットする。なお、ここで取り上げる論理シミュレータ802は一般的な機能レベル命令論理シミュレータとし、処理制御部803と論理モデル804とRAMモデル805を構成要素とするものとして説明を行う。

【0058】次に論理シミュレータ802でセットされたテストベクトル806のテスト命令列を順次シミュレーションすると、特殊な意味を持つ命令列をシミュレーションするとき、その特殊命令列がマイクロ命令動作を伴う場合に、論理シミュレータ802は、インタフェース情報306に現在のRAMモデル805の内容をセットし、マイクロ命令動作を論理エミュレータ装置101に対して要求する。この要求は処理制御部803経由で処理制御プログラム201に伝えられ、セットされたインタフェース情報を論理エミュレータ装置101のインタフェース情報306に転送する。論理シミュレータ802は要求を出した後、引き続きテスト命令列のシミュレーションを再開する。また、インタフェース情報を転送した後、処理制御プログラム201は、論理エミュレータ装置101を動作させ、この時点で論理シミュレータ802と論理エミュレータ装置101が完全に非同期に動作を開始する。次に一定サイクルだけエミュレーションを行うとマイクロ命令動作が完了し、第一の実施の形態と同様の手順で論理エミュレータ装置101は動作を中断する。この時、中断状態をインタフェース情報として論理シミュレータ802に転送するが、非同期に動作している論理シミュレータ802と同期を取るのが同期/非同期制御部217である。

【0059】この同期/非同期制御部217は、論理シミュレータ802がマイクロ命令動作要求を出したときに論理シミュレータ802側で論理エミュレーション完了時刻をあらかじめ予想しているため、論理シミュレータ802がテスト命令列を一定命令数だけシミュレーションすると待ち合わせを行う場合と、一定命令数だけシミュレーションが完了していない場合には、割込み処理として論理シミュレータ802に報告する場合とを制御する。

【0060】報告されたエミュレーション完了報告を処理制御部803が管理し、テスト命令列のシミュレーションを制御する。これら一連の処理を繰り返し行い、論理検証を行う。このように論理シミュレータ802と論

10

20

30

40

50

理エミュレータ装置101が混在する論理検証方式では、インタフェース情報を互いに転送しあい、検証状態の整合性をとりながら、検証途中では完全に非同期に動作し、効率の良い論理検証を行う。また、論理シミュレータ802を接続する場合には、接続プラグインタフェース207を用いて接続するため、接続インタフェースを統一することが可能であれば、どのような論理検証手法であっても接続が可能であり、更に複数の異なる論理検証手段を一括して取扱うことが可能である。

【0061】以上のように第一の実施の形態で説明した通り、検証対象論理構造モデル105に組み込まれた擬似論理モデル304と、テストベクトル303aに組み込まれたテスト制御プログラム305と、これらと情報の授受を行うことで、外部から論理エミュレーションを制御する処理制御プログラム201を組み込むことで、論理エミュレータ装置101の高速性を最大限に利用した高速、且つ、操作性や観測性に優れた対話型の論理検証方式の確立が可能となる。

【0062】また、第二の実施の形態に例示したように、処理制御プログラム201が、たとえばOS等の実行環境に依存しないTcl/Tk等の汎用言語で記述されていることにより、第一の実施の形態で使用したテストベクトル303や処理制御プログラム201をそのまま使用して、検証論理チップ508等の実チップの論理検証が可能な環境を構築することができ、論理モデルから実チップに至るまでの一貫した論理検証が可能となり、論理検証を効率良く行うことが可能である。

【0063】また、第三の実施の形態に例示したように、論理シミュレータ802と論理エミュレータ装置101等のように、検証レベルの異なる論理検証手法を接続し、インタフェース情報を互いに転送することで非同期動作の整合性をとった論理検証が可能であり、たとえば、マイクロプロセッサとその周辺機器の1/0制御デバイス等で構成されるシステム全体等のような、大規模論理の装置全体としての一貫した論理検証が可能である。

【0064】以上本発明者によってなされた発明を実施の形態に基づき具体的に説明したが、本発明は前記実施の形態に限定されるものではなく、その要旨を逸脱しない範囲で種々変更可能であることはいうまでもない。

【0065】

【発明の効果】本発明の論理検証方法によれば、論理チップ全体での論理エミュレーションによるシステム論理検証を短時間に効率よく行うことができる、という効果が得られる。

【0066】また、マンマシンインタフェースを充実させ、論理エミュレーションにおける内部状態の観測性や操作性、さらにはテスト容易性を向上させることができる、という効果が得られる。

【0067】また、論理モデルから実チップに至るまで

の一貫した論理検証を実現することができる、という効果が得られる。

【0068】また、論理検証の環境構築に要する工数や期間を短縮して、論理検証工程におけるコスト削減を実現することができる、という効果が得られる。

【図面の簡単な説明】

【図1】本発明の第一の実施の形態である論理検証方法が実施される情報処理システムの構成の一例を示す概念図である。

【図2】本発明の論理検証方法を実現するためのソフトウェアの構成の一例を示した概念図である。

【図3】本発明の第一の実施の形態である論理検証方法の作用の一例を示すフローチャートである。

【図4】本発明の第二の実施の形態である論理検証方法が実施される情報処理システムの構成の一例を示す概念図である。

【図5】本発明の第二の実施の形態である論理検証方法にて用いられるソフトウェアの構成の一例を示した概念図である。

【図6】本発明の第二の実施の形態である論理検証方法の作用の一例を示すフローチャートである。

【図7】本発明の第三の実施の形態である論理検証方法にて用いられるソフトウェアの構成の一例を示した概念図である。

【図8】本発明の論理検証方法の参考技術である論理エミュレーションシステムの構成の一例を示す概念図である。

【符号の説明】

100…論理エミュレーションシステム、101…論理エミュレータ装置、102…制御装置、103…RAM、104…クロック発生装置、105…検証対象論理構造モデル格納エリア、106…テストベクトル格納エリア、107…制御信号ピン、108…情報処理装置、109…論理エミュレータ制御プログラム、110…制御部、111…論理コンパイラ、112…RAMアクセス部、113…GUI制御部、113…グラフィカルユーザーインタフェース制御部、114…検証対象論理、114a…検証対象論理構造モデル、201…処理制御プログラム、202…GUI制御部、203…プロセス間通信制御部、204…論理エミュレータ制御部、205…テストベクトル制御部、206…データ変換部、207…接続プラグインタフェース、208…トリガーイベント制御部、209…動作制御部、210…RAMアクセス部、211…処理制御部、212…表示部、213…入力制御部、214…ウインドウ制御部、215…設定情報制御部、216…ログデータ採取制御部、217…同期/非同期制御部、300…論理エミュレーションシステム、301…情報処理装置、302…情報処理装置、303…テストベクトル、303a…テストベクトル、304…擬似論理モデル（第2の論理構造モデル）

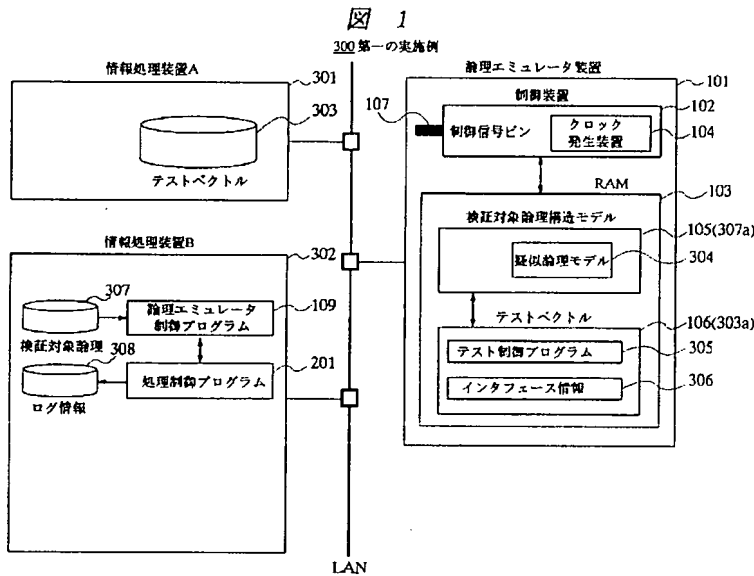
17

ル)、305…テスト制御プログラム、306…インタフェース情報、307…検証対象論理、307a…検証対象論理構造モデル(第1の論理構造モデル)、308…ログ情報、500…論理エミュレーションシステム、501…情報処理装置、502…専用情報処理装置、503…Tcl/Tkインタプリタ、504…ログ情報、505…デバイスドライバ、506…パラレルI/Oインタフェース、507…制御装置、508…検証論理チ*

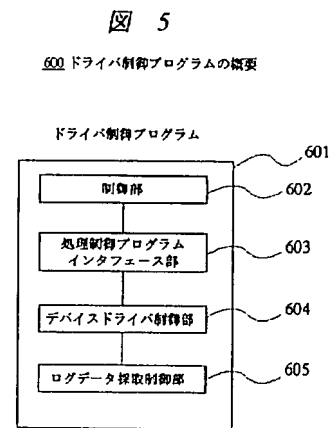
18

* ップ(目的論理装置)、509…RAM、601…ドライバ制御プログラム、602…制御部、603…処理制御プログラムインタフェース部、604…デバイスドライバ制御部、605…ログデータ採取制御部、800…論理エミュレーションシステム、801…情報処理装置、802…論理シミュレータ、803…処理制御部、804…論理モデル、805…RAMモデル、806…テストベクトル。

【図1】

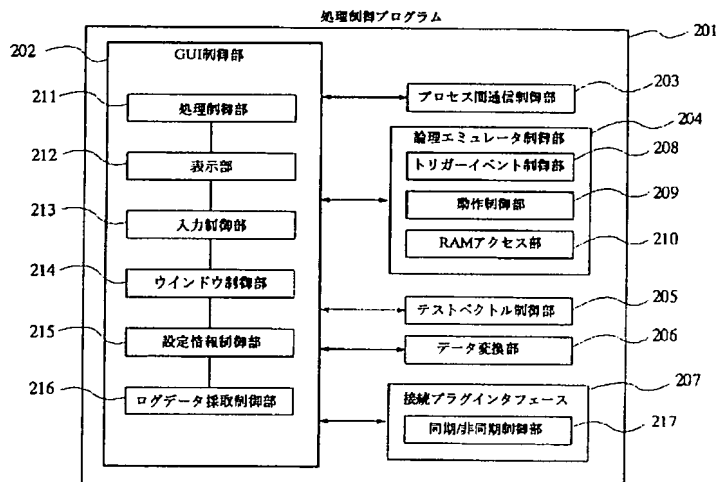


【図5】

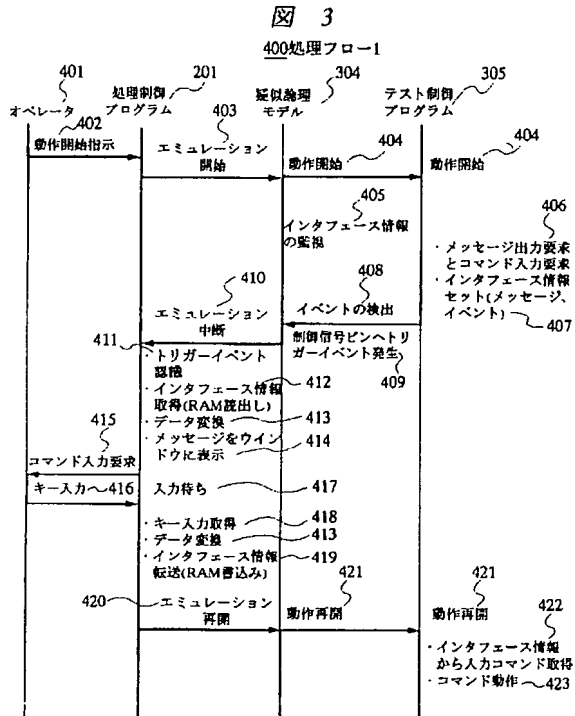


【図2】

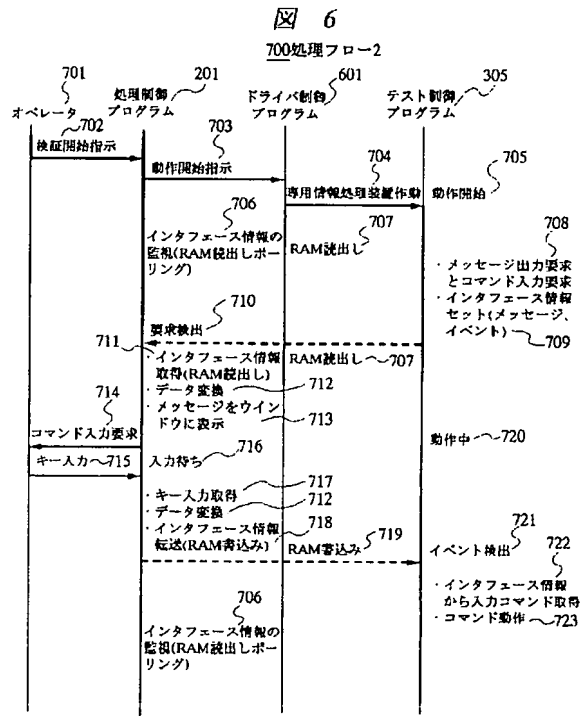
図2
処理制御プログラムの概要



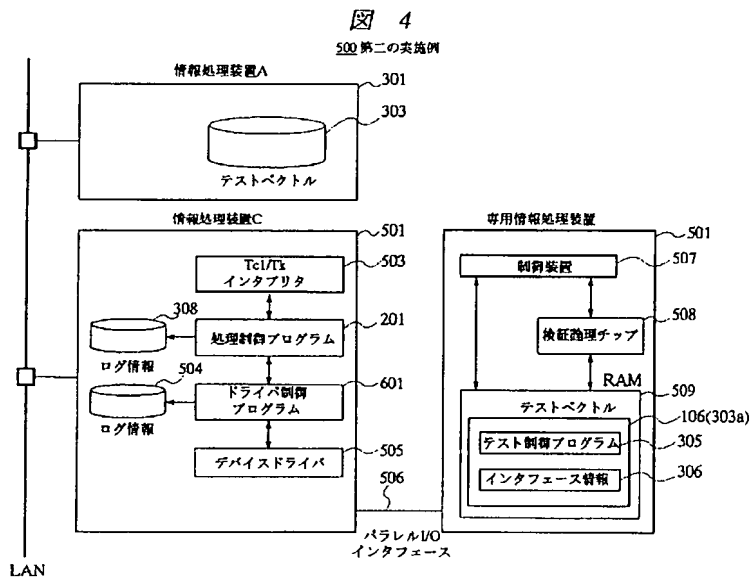
【図3】



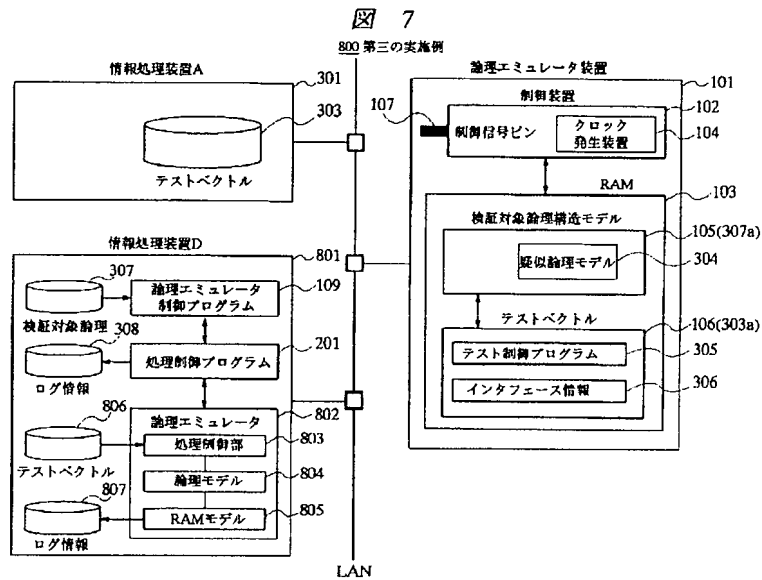
【図6】



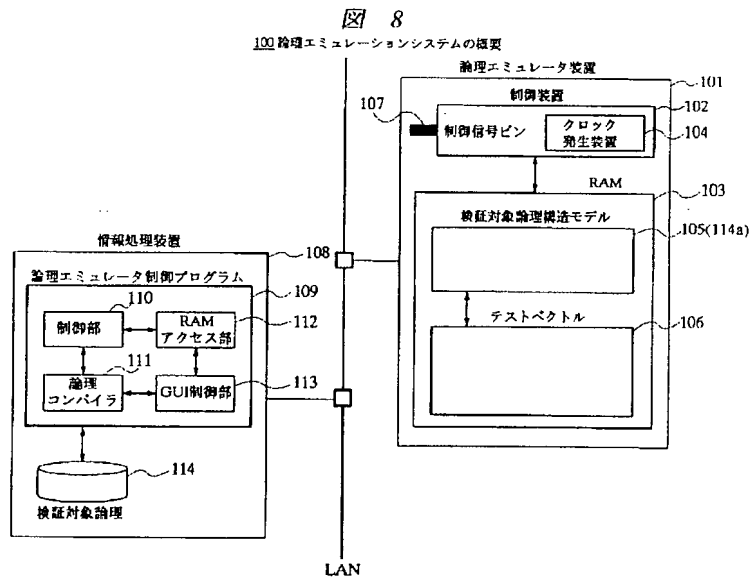
【図4】



【図7】



【図8】



フロントページの続き

(51)Int.Cl.

識別記号

F I
H 0 1 L 21/82

テーマコード(参考)

T

F ターム(参考) 2G032 AA01 AC08 AE07 AE08 AE10
5B046 AA08 BA03 JA05
5B048 AA01 BB02 DD01 DD05 DD15
5F064 HH05 HH09 HH10 HH13 HH14
9A001 BZ05 DZ13 HZ32 JJ49 JZ45